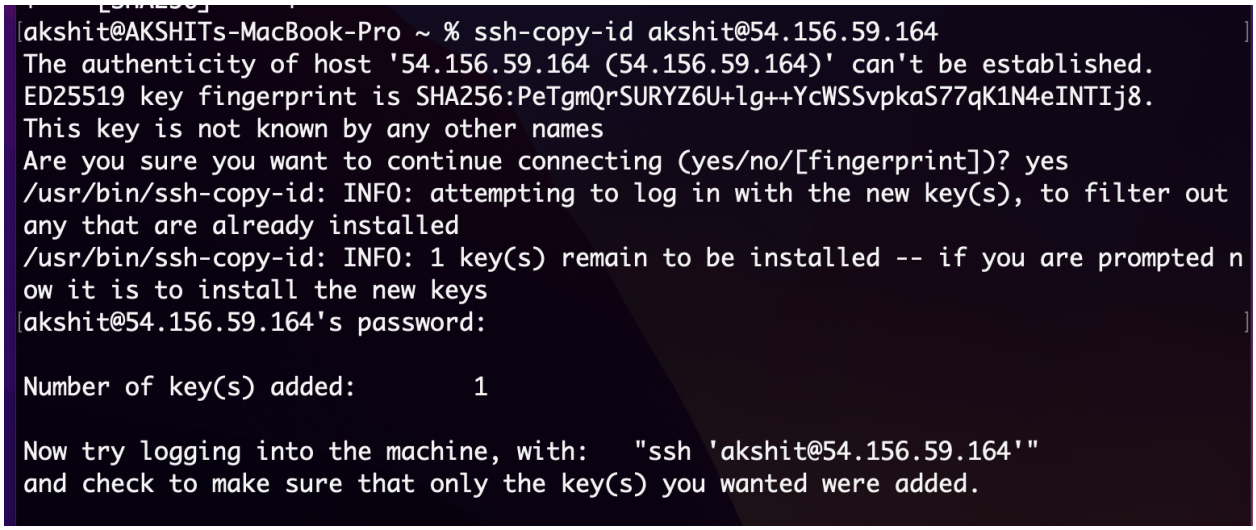
# PROJECT

You are a DevOps engineer at XYZ Ltd. Your company is working on a Java application and wants to automate WAR file artifact deployment so that they don't have to perform WAR deployment on Tomcat/Jetty web containers. Automate Ansible integration with Jenkins CI server so that we can run and execute playbooks to deploy custom WAR files to a web container and then perform restart for the web container.

1. Configure Jenkins server as Ansible provisioning machine
2. Install Ansible plugins in Jenkins CI server
3. Prepare Ansible playbook to run Maven build on Jenkins CI server
4. Prepare Ansible playbook to execute deployment steps on the remote web container with restart of the web container post deployment

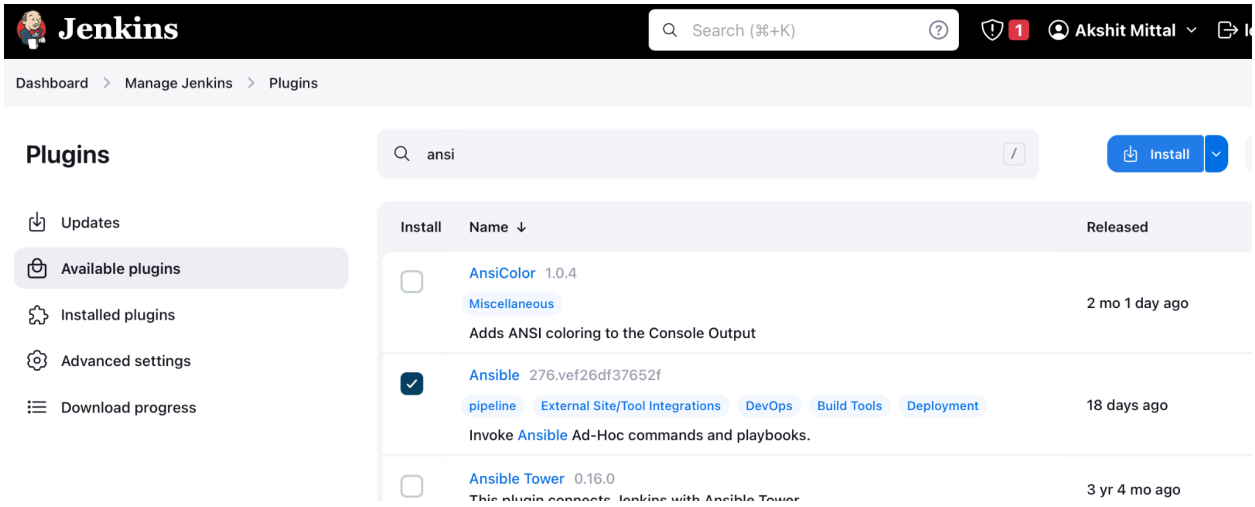
Solution:-

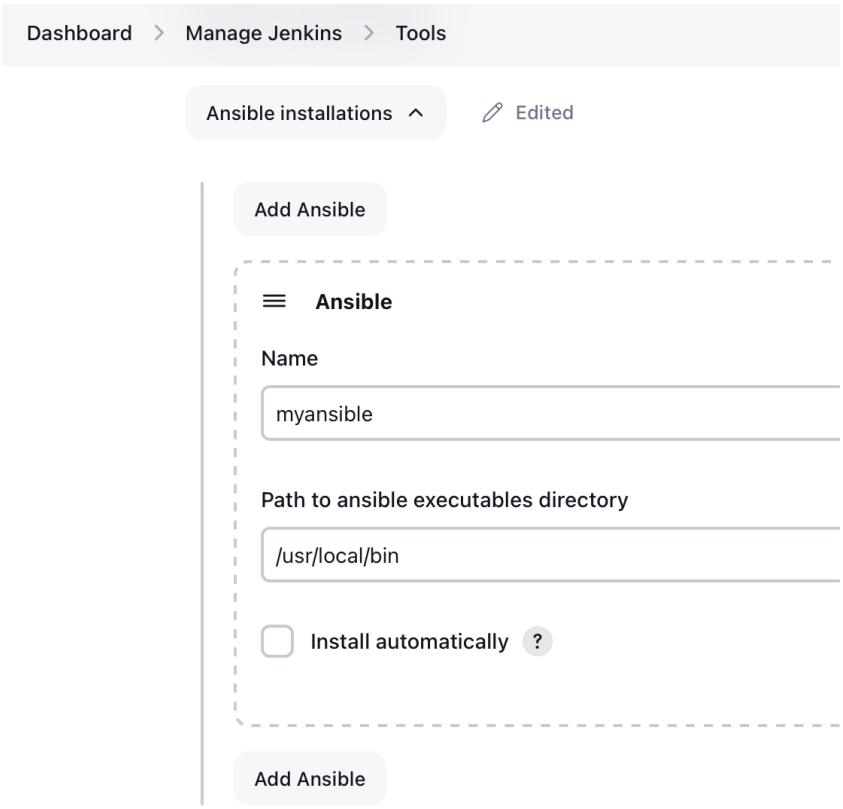
1. On the ACM (ansible controller machine), Connect the master node to the worker nodes by copying ssh-key on worker nodes. Add Public ip of worker nodes in the ansible host file and run the ping command to check the connectivity.



```
akshit@AKSHITs-MacBook-Pro ansible % vim hosts
akshit@AKSHITs-MacBook-Pro ansible % ansible -i hosts webserver -m ping
54.156.59.164 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
akshit@AKSHITs-MacBook-Pro ansible % cat hosts
[webserver]
54.156.59.164
```

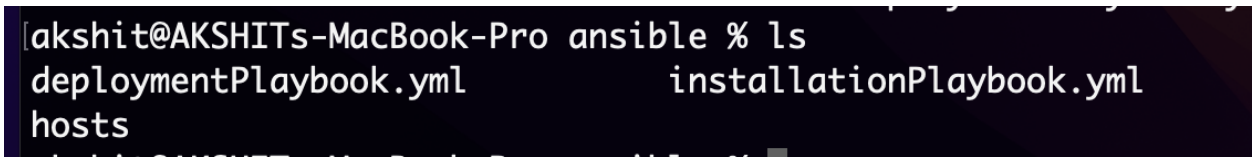
2. On the Controller Machine, Install Java, jenkins. Complete the setup. Start the jenkins service, and go on the plugins section to install Ansible and then → Tools section to add Ansible in tools with ansible-playbook path in the path section of ansible






3. Now our task is to write 2 playbooks- 1 for installation of packages and Another playbook for deployment of code and building the Artifact and deploy on gitHub to be called via jenkins at the time of pipeline run.
   But here,
   First we will try to do it on our Controller machine, if the playbooks are running fine, then we will deploy the playbooks and Inventory file on Github and call via jenkins.

→ So, in the Ansible directory, we will be having 3 files



→ In the Installation Playbook

```
- name: Installation packages playbook
  hosts: webserver
  become: true
  become_user: root
  tasks:
  - name: update the apt-get repo
    command: sudo apt-get update
  - name: Install maven on worker nodes
    package:
     name=maven
     state=present
  - name: Install docker using module
    package:
     name=docker.io
     state=present
  - name: Start docker service
    service:
     name=docker
     state=started
~
~
```

```
akshit@AKSHITs-MacBook-Pro ansible % ansible-playbook -i hosts installationPlaybook.yml

PLAY [Installation packages playbook] ******************************************

TASK [Gathering Facts] *********************************************************
ok: [54.156.59.164]

TASK [update the apt-get repo] *************************************************
changed: [54.156.59.164]

TASK [Install maven on worker nodes] *******************************************
ok: [54.156.59.164]

TASK [Install docker using module] *********************************************
changed: [54.156.59.164]

TASK [Start docker service] ****************************************************
ok: [54.156.59.164]

PLAY RECAP *********************************************************************
54.156.59.164              : ok=5    changed=2    unreachable=0    failed=0    skipped=0    re
scued=0    ignored=0
```

The playbook is executing successfully

→ In the deployment playbook

```
- name: Playbook to clone, build, deploy the java code
  hosts: webserver
  become: true
  become_user: root
  tasks:
  - name: clone the repo with java code on every worker node
    git:
     repo=https://github.com/Sonal0409/DevOpsCodeDemo.git
     dest=/tmp/devopscode-repo
  - name: Maven to build the code
    command: chdir=/tmp/devopscode-repo mvn package
  - name: From target folder copy the addressbook.war in current directory
    copy:
     src=/tmp/devopscode-repo/target/addressbook.war
     dest=/tmp/devopscode-repo
     remote_src=yes
  - name: Build the dockerfile
    command: chdir=/tmp/devopscode-repo  docker build -t mycodeimage .
  - name: Run the image to deploy the application on tomcat container
    command: docker run -d -P mycodeimage
~
~
~
```

- We have cloned the git repo in the folder
- Using the maven tool, we build the code via package module
- We saw that the build executable file with extension ".war" was not present in the same folder but in the target folder names addressbook.war, so we copied the file into the main folder
- We run the docker command to build the image of our file
- We run the image to build the container running on the server as seen in the screenshots below.

```
akshit@AKSHITs-MacBook-Pro ansible % ansible-playbook -i hosts deploymentPlaybook.yml

PLAY [Playbook to clone, build, deploy the java code] **************************************

TASK [Gathering Facts] *********************************************************************
ok: [54.156.59.164]

TASK [clone the repo with java code on every worker node] **********************************
changed: [54.156.59.164]

TASK [Maven to build the code] *************************************************************
changed: [54.156.59.164]

TASK [From target folder copy the addressbook.war in current directory] ********************
changed: [54.156.59.164]

TASK [Build the dockerfile] ****************************************************************
changed: [54.156.59.164]

TASK [Run the image to deploy the application on tomcat container] *************************
changed: [54.156.59.164]

PLAY RECAP *********************************************************************************
54.156.59.164              : ok=6    changed=5    unreachable=0    failed=0    skipped=0    re
scued=0    ignored=0
```

The .war deployable file has been copied into the main folder

```
akshit@AKSHITs-MacBook-Pro ansible % ansible -i hosts webserver -m command -a "ls /tmp/devopscode-repo"
54.156.59.164 | CHANGED | rc=0 >>
CICD-pipeline-complete
Jenkinsfile
Newfiletraigger
README.md
addressbook.war
addressbook_screenshot.png
build.properties
build.xml
demofile
deployment.yml
dockerfile
jenkinsfile2
kubedeploy.yml
newcommit
newdemofile
pom.xml
src
target
```

The image has been built successfully and the container is running successfully.

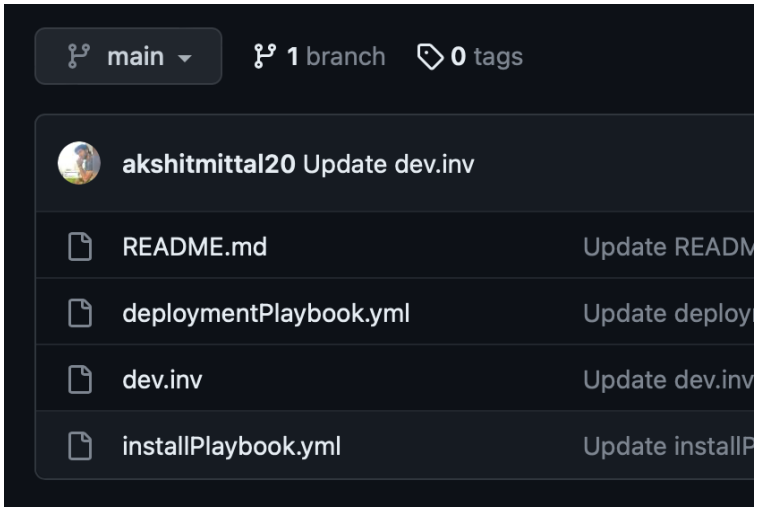
```
akshit@AKSHITs-MacBook-Pro ansible % ansible -i hosts webserver -m command -a "sudo docker images"
54.156.59.164 | CHANGED | rc=0 >>
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
mycodeimage   latest    19313eeec778   13 minutes ago   478MB
tomcat        9         3d0efbe5f648   7 days ago       462MB
```

```
akshit@AKSHITs-MacBook-Pro ansible % ansible -i hosts webserver -m command -a "sudo docker ps -a"
54.156.59.164 | CHANGED | rc=0 >>
CONTAINER ID   IMAGE         COMMAND             CREATED         STATUS         PORTS
                NAMES
3b8fa98352cb   mycodeimage   "catalina.sh run"   8 minutes ago   Up 8 minutes   0.0.0.0:32768->8080/tcp, :::
32768->8080/tcp   keen_poincare
```

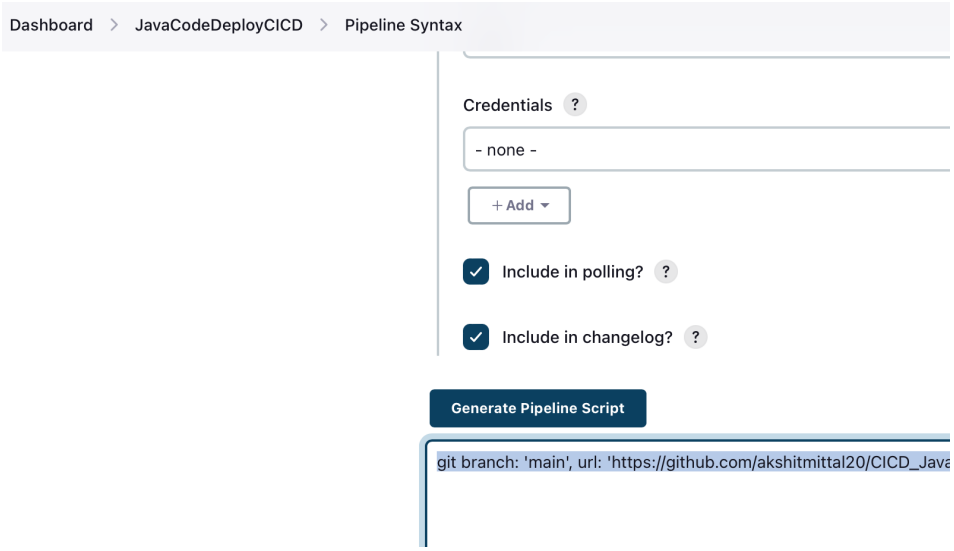
Now, we will do this via automation using Jenkins.

4. We will copy all 3 files on the git repo , which will be then called in the jenkins pipeline.

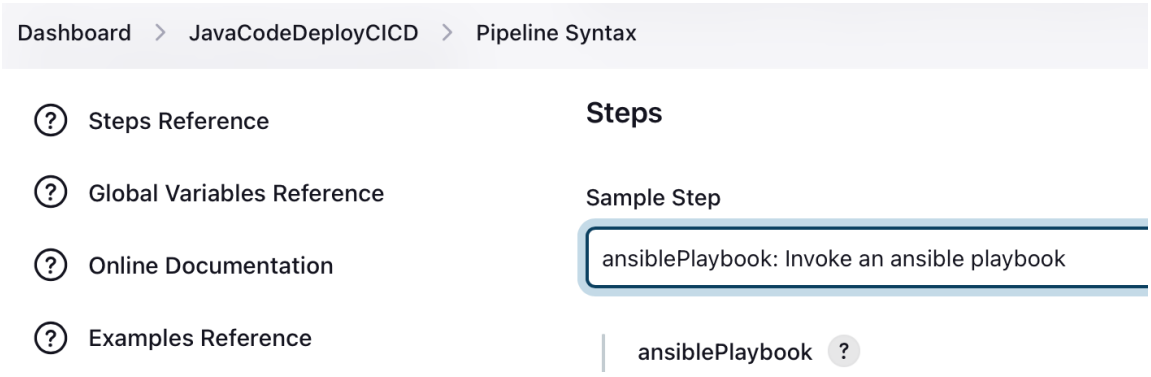
```
akshit@AKSHITs-MacBook-Pro ansible % ls
deploymentPlaybook.yml          hosts                           installationPlaybook.yml
akshit@AKSHITs-MacBook-Pro ansible % cat hosts
[webserver]
54.156.59.164
```



→ We can take the help of pipeline syntax feature to generate the pipeline

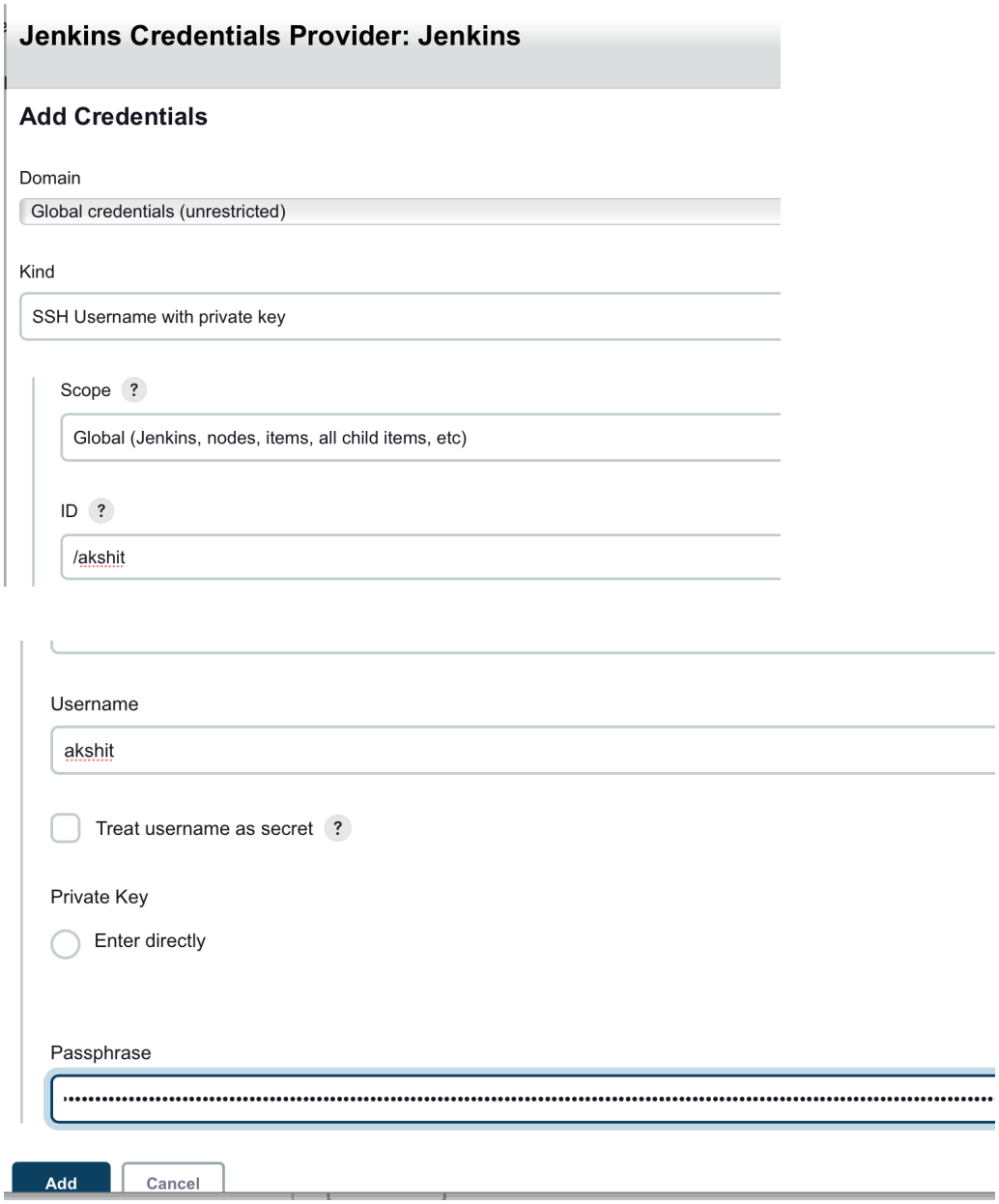



→ Use ansiblePlaybook; Invoke module for generating ansible playbook syntax



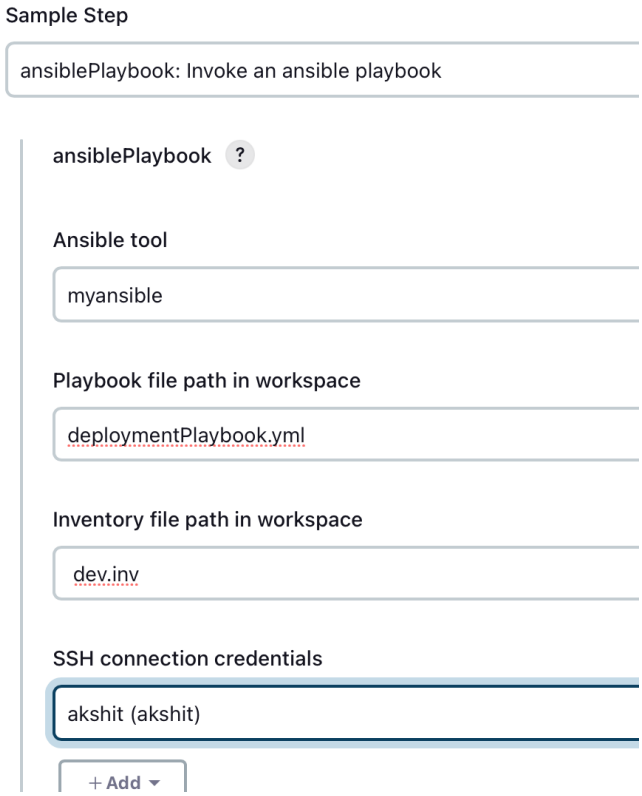

→ Give credentials of controller machine username and ssh-key in the ssh credentials provider tab. Select Kind as “SSH username with private key” → Give your username in which ssh key is generated and ssh private key in the passphrase tab.

You can find ssh private key in controller machine using command #cat ~/.ssh/id_rsa

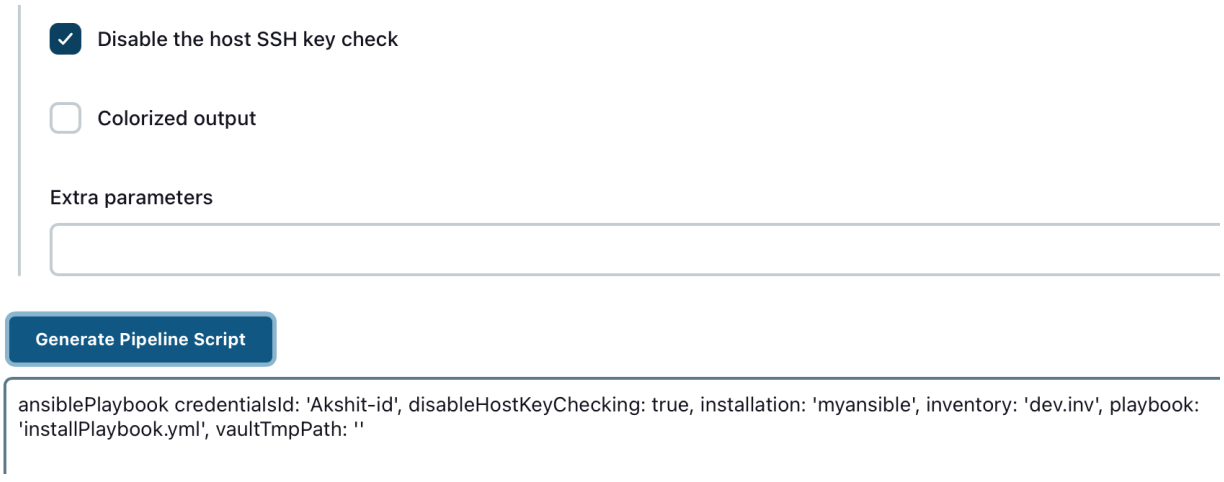



```
akshit@AKSHITs-MacBook-Pro ~ % cat ~/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAtnVDQiPQfR++ibDKpkJRD4mStGSxSE2Sp5gLUHzyz0BLdkHSp0z/
U56OeAOkvRJkqkDItIgI9UiH34/Al5ITI1iL2a8Qte2YeE9WjGgtbh9PmyGNF/KlljYEd1
qKryRcbEOpUFfFo3Ph1M/FeISrIxVe9riw2aKNScvJ/V1i+QWQxdiErFLd2bTVGwdNRVFA
j59bF8Dsl+4kLxndv0BLz1R3JALp5d0s7YAWRC5lhmWX6IOMIPN7aLKs+/3Ovbp/yOVPiR
NCuYEpOZojHE9+llCou+iIbFExrnRifgIQivVjdGFNk+gss+4hn90Zm4KwMhwk8wasyb2+
I4O+Y4B7ObpC/YroRfSVc3RWi1kxLWv0pV3t8nsONESkTtxQMs+OkYclX9SR1ri1YGmy8L
```

→ give name of repo file, for ip address, and installation playbook




→ Disable the host key check. And generate the script



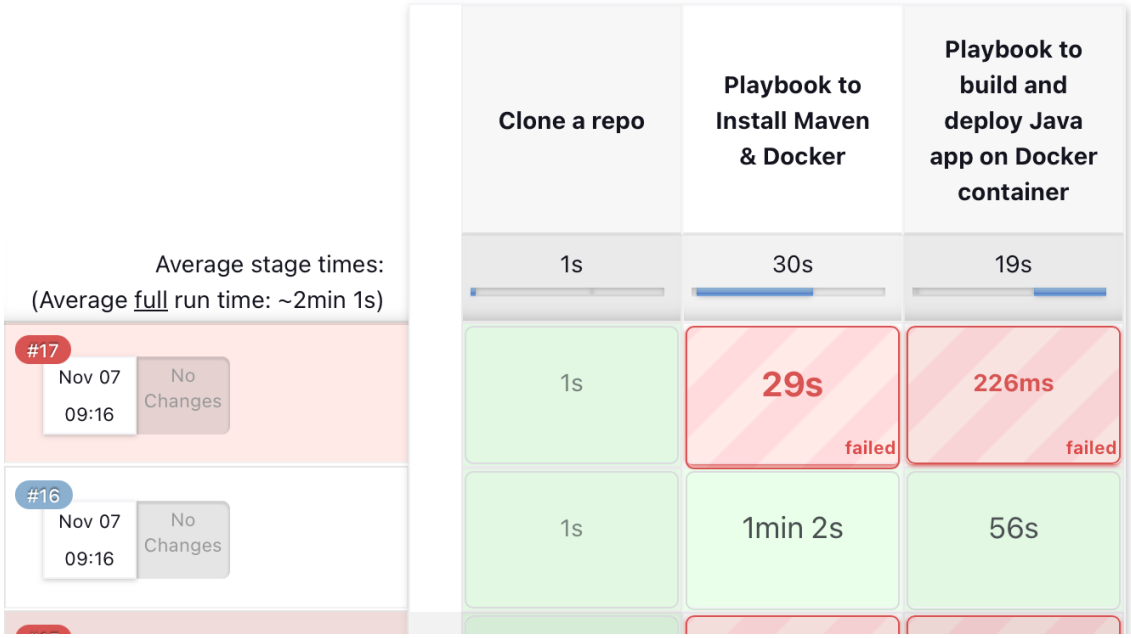

→ Similarly do the same steps for deployment playbook

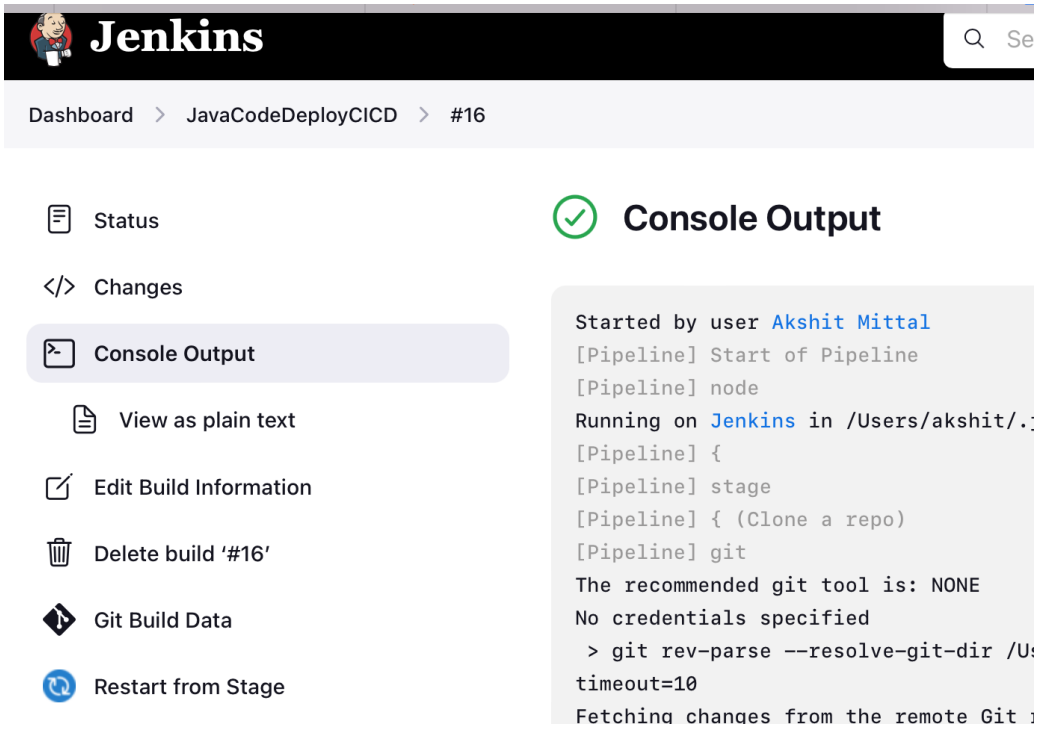
→ Finally your pipeline will look like this

```
pipeline {
    agent any

    stages {
        stage('Clone a repo') {
            steps {
                git branch: 'main', url: 'https://github.com/akshitmittal20/CICD_JavaApplicationDeployment_Ansible_Docker_Jenkins.git'
            }
        }

        stage('Playbook to Install Maven & Docker') {
            steps {
                ansiblePlaybook credentialsId: 'akshit',
                disableHostKeyChecking: true,
                installation: 'myansible',
                inventory: 'dev.inv',
                playbook: 'installPlaybook.yml'
            }
        }

        stage('Playbook to build and deploy Java app on Docker container') {
            steps {
                ansiblePlaybook credentialsId: 'akshit',
                disableHostKeyChecking: true,
                installation: 'myansible',
                inventory: 'dev.inv',
                playbook: 'deploymentPlaybook.yml'
            }
        }
    }
}
```

→ Generate the output
Click on Build now and generate the output

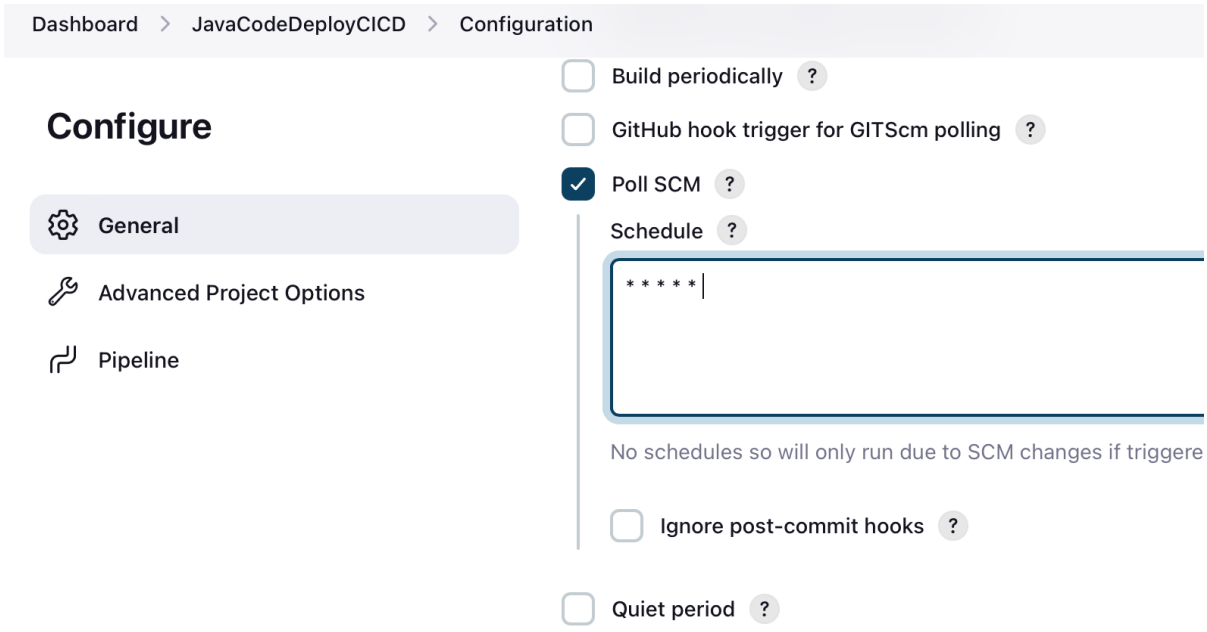
## Stage View






Pipeline is Successfully running.

For continuous deployment automation setup , we can set up the triggers in Jenkins, and it will check after sometime if any new commit is there, and pipeline will get triggered, via function Poll SCM.




==========